Quality Metrics for Business Process Models

Irene Vanderfeesten[1], Jorge Cardoso[2], Jan Mendling[3], Hajo A. Reijers[1], Wil van der Aalst[1]

[1] Technische Universiteit Eindhoven, The Netherlands; [2] University of Madeira, Portugal; [3] Vienna University of Economics and Business, Austria.

SUMMARY

In the area of software engineering, quality metrics have shown their importance for good programming practices and software designs. A design developed by the help of these metrics (e.g. coupling, cohesion, complexity, modularity and size) as guiding principals is likely to be less error-prone, easy to understand, maintain, and manage, and is more efficient. Several researchers already identified similarities between software programs and business process designs and recognized the potential of quality metrics in business process management (Cardoso, Mendling, Neuman & Reijers, 2006; Gruhn & Laue, 2006; Latva-Koivisto, 2001). This chapter elaborates on the importance of quality metrics for business process modeling. It presents a classification and an overview of current business process metrics and it gives an example of the implementation of these metrics using the ProM tool. ProM is an analysis tool, freely available, that can be used to study process models implemented in more than eight languages.

INTRODUCTION

Key in many instances of innovation is the transfer of information and understanding developed in one discipline to the other (Kostoff, 1999). A prime example is workflow management itself, a technology based on the importation of process models from manufacturing operations into administrative work. In this chapter we embark on further opportunities for knowledge transfer to the field of process modeling and workflow management, in particular from software engineering.

In the mid-1960's software engineers started to use metrics to characterize the properties of their code. Simple count metrics were designed to be applied to programs written in languages such as C++, Java, FORTRAN, etc. Various of these metrics provided a good analysis mechanism to assess the quality of the software program design. Since there is a strong analogy between software programs and business processes, as previously argued in (Reijers & Vanderfeesten, 2004; Gucegliogl u & Demiros, 2005), we believe that software metrics, such as coupling, cohesion, and complexity, can be revised and adapted to analyze and study a business process' characteristics.

A business process model, regardless whether it is modeled in e.g. BPEL, EPC, BPMN or Petri Nets, exhibits many similarities with traditional software programs. A software program is usually partitioned into modules or functions (i.e. activities), which take in a group of inputs and provide some output. Similar to this compositional structure, a business process model consists of activities, each of which contains smaller steps (operations) on elementary data elements (see Table 1). Moreover, just like the interactions between modules and functions in a software pro-

gram are precisely specified, the order of activity execution in a process model is predefined using logic operators such as sequence, splits and joins.

In this chapter we elaborate on the transfer and adaptation of quality metrics from the software engineering domain to business processes. First, we introduce a general outline on software engineering metrics. Next, an overview is given of the current status in business process metrics, adopting a widely used classification from software engineering. Finally, we elaborate on some practical applications of these business process metrics and present our conclusions and a look into the future of business process metrics.

Table 1: Similarities between software programs and business processes

Software program	Business process
Module/Class	Activity
Method/Function	Operation
Variable/Constant	Data element

METRICS IN THE SOFTWARE ENGINEERING DOMAIN

In the area of software engineering a wide variety of software quality metrics has been developed. The main purpose of software quality metrics is to obtain program designs that are better structured. Some of the most important advantages of a structured design are, as pointed out in (Conte, Dunsmore & Shen, 1986), that (i) the overall program logic is easier to understand for both the programmers and the users and (ii) the identification of the modules is easier, since different functions are performed by different modules, which makes the maintenance of the software program easier. According to (Conte, Dunsmore & Shen, 1986; Shepperd 1993; Troy & Zweben 1981) the quality of a design is related to five design principles:

Coupling—Coupling is measured by the number of interconnections among modules. Coupling is a measure for the strength of association established by the interconnections from one module of a design to another. The degree of coupling depends on how complicated the connections are and on the type of connections. It is hypothesized that programs with a high coupling will contain more errors than programs with lower coupling.

Cohesion—Cohesion is a measure of the relationships of the elements within a module. It is also called module strength. It is hypothesized that programs with low cohesion will contain more errors than programs with higher cohesion.

Complexity—A design should be as simple as possible. Design complexity grows as the number of control constructs grows, and also as the size—in number of modules—grows. The hypothesis is that the higher the design complexity the more errors the design will contain.

Modularity—The degree of modularization affects the quality of a design. Over-modularization is as undesirable as under-modularization. The hypothesis is that low modularity generally relates to more errors than higher modularity.

Size—A design that exhibits large modules or a deep nesting is considered undesirable. It is hypothesized that programs of large size will contain more errors than smaller programs.

In literature coupling and cohesion are generally considered to be the most important metrics for software quality, although researchers do not agree on their relative importance. In (Shepperd 1993; Troy & Zweben 1981), results of analyses are presented that indicate that coupling is the most influential of the design principles under consideration. However, in (Myers, 1978) cohesion and coupling are consid-

ered as equally important. Also, complexity and size are seen as a good quality metric for software program designs (Troy & Zweben, 1981).

In addition, various researchers carried out studies to gather empirical evidence that quality metrics do indeed improve the quality of a software design. Bieman and Kang, in particular, have shown examples how cohesion metrics can be used to restructure a software design (Bieman & Kang, 1998; Kang & Bieman, 1996; Kang & Bieman 1999). Also, in (Selby & Basili, 1991) evidence is presented that low coupling and high strength (cohesion) are desirable. By calculating coupling/strength ratios of a number of routines in a software library tool it was found that routines with low coupling/strength ration had significantly more errors than routines with high coupling/strength ratio. In (Card, Church & Agresti, 1986), a number of Fortran modules from a National Aeronautics and Space Administration project were examined. Fifty percent of the high strength (high cohesion) modules were fault free, whereas only 18 percent of low strength modules were fault free. No relationship was observed between fault rate and coupling in this study. Finally, (Shen, Yu, Thebaut & Paulsen, 1985) have discovered that, based on their analysis of three software program products and their error histories, simple metrics such as the amount of data (size) and the structural complexity of programs may be useful in identifying those modules most likely to contain errors.

QUALITY METRICS IN THE WORKFLOW DOMAIN

Because of the similarities between software programs and workflow processes, explained in the introduction, the application of similar quality metrics to the workflow field is worth investigation. We have conducted a literature review on business process metrics and found that, despite the vast literature on software engineering metrics, there is not much research on business process metrics available yet. Although some researchers suggest using software metrics to evaluate business process designs (Baresi et al, 1999), the number of publications on concrete metrics and applications in the business process domain is still small and only of a very recent date. In this section, the existing "state-of-the-art" in business process metrics is summarized using the same classification as in software engineering.

Coupling

Coupling measures the number of interconnections among the modules of the model. As such, it is highly related to degree and density metrics in (social) network analysis (see e.g. Brandes and Erlebach, 2005). The application of these measurements is straight forward if the process model is available in a graph-based notation. The average degree, also called coefficient of connectivity in (Latva-Koivisto, 2001), refers to the average number of connections that a node has with other nodes of the process graph. In contrast to that, the density metric relates the number of available connections to the number of maximum connections for the given number of nodes. The density metric was used in a survey as a predictor for errors in business process models in (Mendling, 2006) with some mixed results. While there was actually a connection between density and errors, the explanatory power of this metric was found to be limited. An explanation for that might be that density is difficult to compare for models of different size since the maximum number of connections grows quadratic. It appears that the average degree of nodes in a business process could be better suited to serve as a quality metric.

Moreover, Reijers and Vanderfeesten (Reijers & Vanderfeesten, 2004) also developed a similar coupling metric counting the overlap of data elements for each pair of activities using a static description of the product represented by a Product Data Model (PDM) (van der Aalst, 1999; Reijers, 2003; Reijers, Limam Mansar, & van der

Aalst, 2003). Two activities are 'coupled' if they contain one or more common data elements. To calculate the coupling value for a business process the activities are selected pairwise and the number of 'coupled' pairs is counted. Finally, the mean is determined based on the total number of activities. The outcome always lies between 0 and 1. This data oriented coupling metric is complemented with a cohesion metric, which is described in the next section.

However, all of these coupling metrics do not yet deal with how complicated the connections are as suggested in the definition of coupling. A weighted coupling metric, with different weights for the XOR, OR, and AND connectors, is part of our current research.

Cohesion

Cohesion measures the coherence within the parts of the model. So far, there is only one paper on a cohesion metric for business processes available. Reijers and Vanderfeesten (Reijers & Vanderfeesten, 2004) developed a cohesion metric for workflow processes which looks at the coherence within the activities of the process model. Similar to their coupling metric this cohesion metric also focuses on the information processing in the process and takes a data oriented view. For each activity in the process model the total cohesion is calculated by multiplying the information cohesion and the relation cohesion of the activity. Finally, a cohesion value for the whole process is determined by taking the mean of all activity cohesion values (i.e. adding up all cohesion values and dividing it by the number of activities). The value for this cohesion metric always lies between 0 and 1. This data oriented cohesion metric is complemented with a coupling metric, which is described in the previous section. The combination of these two metrics, as proposed by (Selby & Basili, 1991), gives a coupling-cohesion ratio which supports the business process designer to select the best (low coupling, high cohesion) design among several alternatives (Reijers & Vanderfeesten, 2004).

Complexity

Complexity measures the simpleness and understandability of a design. In this area most of the research on business process metrics has been done (Cardoso, Mendling, Neumann & Reijers, 2006; Gruhn & Laue, 2006, Latva-Koivisto, 2001). For instance, both (Gruhn & Laue, 2006) and (Cardoso, Mendling, Neumann & Reijers, 2006) consider the adaptation of McCabe's cyclometric number as a complexity metric for business processes. This complexity metric directly measures the number of linearly independent paths through a program's source code. In practice, the industry interpretation of McCabe's cyclomatic complexity thresholds are the following (Frappier, Matwin, & Mili, 1994): from 1 to 10, the program is simple; from 11 to 20, it is slightly complex; from 21 to 50, it is complex; and above 50 it is untestable.

In (Cardoso, 2005a) the Control-Flow Complexity (CFC) metric is defined, which is also derived from software engineering. The CFC metric evaluates the complexity introduced in a process by the presence of XOR-split, OR-split, and AND-split constructs. For XOR-splits, the control-flow complexity is simply the fan-out of the split, i.e. $CFC_{XOR\text{-}split}(a)$= fan-out(a). For OR-splits, the control-flow complexity is 2^n-1, where n is the fan-out of the split. i.e. $CFC_{OR\text{-}split}(a)= 2^{fan\text{-}out(a)}\text{-}1$. For an AND-split, the complexity is simply 1, i.e. $CFC_{AND\text{-}split}(a)= 1$. Mathematically, the control-flow complexity metric is additive. Thus, it is very easy to calculate the complexity of a process, by simply adding the CFC of all split constructs. The greater the value of the CFC, the greater the overall architectural complexity of a process. This metric was evaluated in terms of Weyuker's properties to guarantee that it qualifies as a good

and comprehensive one (Cardoso, 2006). To test the validity of the metric, an experiment has been carried out for empirically validation (Cardoso, 2005b). It was found that the CFC metric is highly correlated with the control-flow complexity of processes. This metric can, therefore, be used by business process analysts and process designers to analyze the complexity of processes and, if possible, develop simpler processes.

Other researchers, for instance (Latva-Koivisto, 2001), also propose graph complexity metrics, such as the Coefficient of Network Complexity (CNC) or the Complexity Index (CI), to evaluate business processes. In general, Cardoso et al (Cardoso, Mendling, Neumann & Reijers, 2006) have identified three different types of business process complexity: (i) computational complexity, (ii) psychological complexity, and (iii) representational complexity.

Modularity

Modularity measures the degree to which a design is split op into several modules. Our literature review has not provided any business process metric that measures the modularity of a business process design. This is no surprise regarding the fact that activites are most often treated as black boxes in business process modeling.

Size

Size simply measures how big a model is. The size of a business process model can be determined using a measure similar to the number of Lines of Code (LOC) from software engineering metrics. The LOC metric in software engineering has been used for years with a significant success rate (Jones 1986). Cardoso et al., Gruhn & Laue and Latva-Koivisto (Cardoso, Mendling, Neumann & Reijers, 2006; Gruhn & Laue, 2006; Latva-Koivisto, 2001) all propose to count the number of activities to establish a measure for size.

While this size metric is very simple, it is very important to complement other forms of process analysis. For example, the control-flow complexity of a process can be very low while its activity complexity can be very high. For example, a sequential process that has a thousand activities has a control-flow complexity of 0, whereas its activity complexity is 100.

From the "state-of-the-art" in business process metrics, we conclude that this field of research is just at its start and that there is a lot of potential for further development of business process metrics. This classification, which was adopted from the software engineering field, is not yet very precise. For instance, Mendling uses a coupling metric as means to calculate complexity (Mendling, 2006) and Latva-Koivisto, Gruhn & Laue, and Cardoso et al. also use size as a measure for complexity (Cardoso, Mendling, Neumann & Reijers, 2006; Gruhn & Laue, 2006; Latva-Koivisto, 2001). Perhaps, this classification of business process metrics should be revised in the future when this area is more mature.

Moreover, we observe that the values for each metric do not yet have a clear meaning, e.g. when the value for coupling for a certain business process model is 0.512 we do not yet know just from the number whether this is high or low, or good or bad. According to (Cardoso, 2005a) it can take several years and a lot of empirical research before such a number really makes sense and quantifies the design in a proper way. Despite this, business process metric analysis in the current situation still gives the designer some insights and guidance on the quality of the design. Moreover, we believe in the potential of these metrics and their importance for business process design in the future.

APPLICATION

Besides the theoretical overview of business process metrics which was provided in the previous sections, we would also like to give some insight in the practical application of these metrics so far. Because this area emerged only recently, there are only a few applications available, while a lot of new research is ongoing at the moment of writing this chapter.

The practical applications that we present here mainly have two directions. First of all, we look at the capabilities of a set of metrics for predicting errors (i.e. we investigate whether there is a relationship between the value of the metrics and the presence of errors in the business process model). Secondly, we present the early implementation of a tool that supports designing of business process models guided by these metrics.

Prediction of error probability based on metrics

Among our hypotheses on the use of business process metrics we state that business process models which are designed using the business process metrics contain less errors, are easier to understand and maintain. A first step made towards the empirical validation of this hypothesis is made in a quantitative analysis about the connection between simple metrics and error probability in the SAP reference model (Mendling et al, 2006a; Mendling et al, 2006b). The SAP reference model is a collection of EPC business process models that was meant to be used as a blueprint for rollout projects of SAP's ERP system (Keller & Teufel, 1998). It reflects Version 4.6 of SAP R/3 which was marketed in 2000. The extensive database of this reference model contains almost 10,000 sub-models, about 600 of them are EPC business process models.

The survey reported in Mendling et al (2006a) includes two parts: the verification of relaxed soundness (which is a minimal correctness criterion for business process models) and the prediction of error probability based on statistic methods. The *verification of relaxed soundness* revealed that about 6 % of the EPC models (34 of 604) contained errors such as e.g. deadlocks. This result on its own emphasizes the need for verification tools in business process modeling projects.

In the second part, the authors investigate the question whether errors appear by chance in a process model, or if there is some way to use business process metrics to predict the error probability. The hypothesis behind this research is that large and complex models are more likely to contain errors, basically because the human modeler is more likely to loose the overview of all interrelations represented in the model. The authors use a set of simple metrics related to size of the models as input to a logistic regression model, i.e. a statistical model to predict occurrence or non-occurrence of an event. The event in this context is whether the process model has an error or not. The results show that these simple metrics are indeed suitable to predict the error probability. In particular, it appears that a higher number of join-connectors is most strongly connected with an increase in error probability.

This survey illustrates one promising application of business process metrics. Still, there is further research needed to identify more elaborate and sophisticated metrics. Moreover, there is a need for further empirical investigation in order to establish an understanding of when a threshold value of a certain metrics indicates bad design in terms of maintainability or likely error proneness.

The ProM tool

In recent years ProM has emerged as a broad and powerful process analysis tool, supporting all kinds of analysis related to business processes (van Dongen et al,

2005). In contrast to many other analysis tools the starting point was the analysis of real processes rather than modeled processes, i.e., using *process mining* techniques ProM attempts to extract non-trivial and useful information from so-called "event logs". Moreover, ProM also allows for the calculation of several quality metrics as will be illustrated later in this section.

Traditionally, most analysis tools focusing on processes are restricted to *model-based analysis*, i.e., a model is used as the starting point of analysis. For example, a purchasing process can be modeled using EPCs and verification techniques can then be used to check the correctness of the protocol while simulation can be used to estimate performance aspects. Such analysis is *only useful if the model reflects reality*. Process mining techniques use event logs as input, i.e., information recorded by systems ranging from enterprise information systems to web services. Hence the starting point is not a model but the observed reality. Therefore, we use the phrase "real process analysis" to position process mining with respect to classical model-based analysis. The widespread use of information systems, e.g., systems constructed using ERP, WFM, CRM, SCM, and PDM software, resulted in the omnipresence of vast amounts of event data. Events may be recorded in the form of audit trails, transactions logs, or databases and may refer to patient treatments, order processing, claims handling, trading, travel booking, etc.

Figure 1 is used to explain the different types of process analysis supported by ProM. First of all, it is relevant to note that when studying business processes one can look at models (lower left corner) or study the observed behavior (lower right corner).

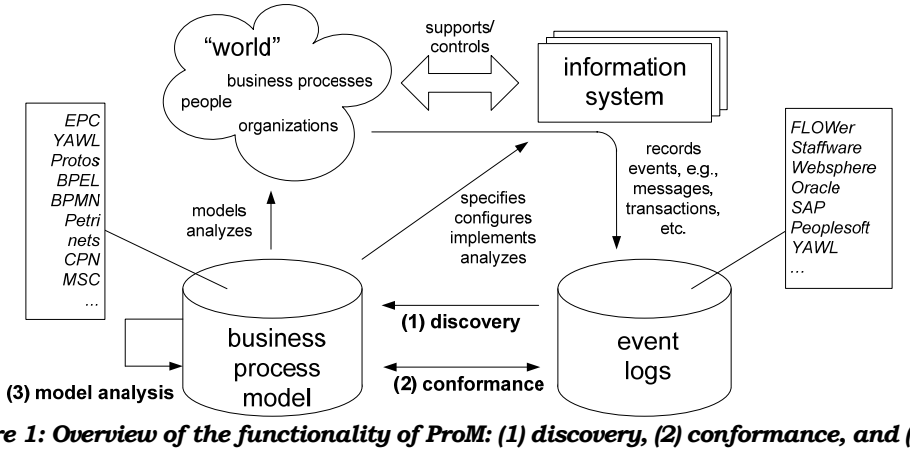

Figure 1: Overview of the functionality of ProM: (1) discovery, (2) conformance, and (3) model analysis

Using process mining it is possible to automatically derive process models using process mining techniques (van der Aalst, Weijters & Maruster, 2004). ProM offers many process discovery techniques. The result may be a Petri net, EPC, or YAWL model. Figure 1 shows some of the modeling notations supported by ProM and also mentions some of the products that provide event logs in a format usable by ProM. Also the list of languages suggests a focus on pure process models, discovery does not need to be limited to control-flow and may also include temporal, resource, data, and organizational aspects.

If a model is already given, the information stored in logs can be used to check conformance, i.e., how well do reality and the model fit together. This can be seen as another quality dimension. Conformance checking requires, in addition to an event log, some a-priori model. This model may be handcrafted or obtained through proc-

ess discovery. Whatever its source, ProM provides various ways of checking whether reality conforms to such a model (Rozinat & van der Aalst, 2006). For example, there may be a process model indicating that purchase orders of more than one million Euro require two checks. Another example is the checking of the so-called "four-eyes principle". Conformance checking may be used to detect deviations, to locate and explain these deviations, and to measure the severity of these deviations.

Last but not least, ProM also provides various ways of model analysis. ProM offers various plug-ins to analyze the correctness of a model, e.g., soundness and absence of deadlocks. For example, it is possible to load the SAP reference model expressed in terms of EPCs into ProM and analyze it using reduction rules or invariants. ProM also allows for the verification of a variety of modeling languages (e.g., BPEL, Staffware, etc.) using a mapping onto Petri nets. Besides model verification, ProM also allows for the calculation of various other quality metrics, e.g., cohesion and coupling, complexity, size, etc. Given the topic of this chapter, we elaborate on these metrics.

Complexity and Size in ProM

In order to study the complexity of process models we have developed several plug-ins for the ProM framework. As stated previously, ProM provides various ways of model analysis, such as soundness and absence of deadlocks. The newly developed plug-ins target the analysis of the quality of process designs. Figure 2 shows one of the plug-ins analyzing the complexity, coupling, and size of an EPC process model.

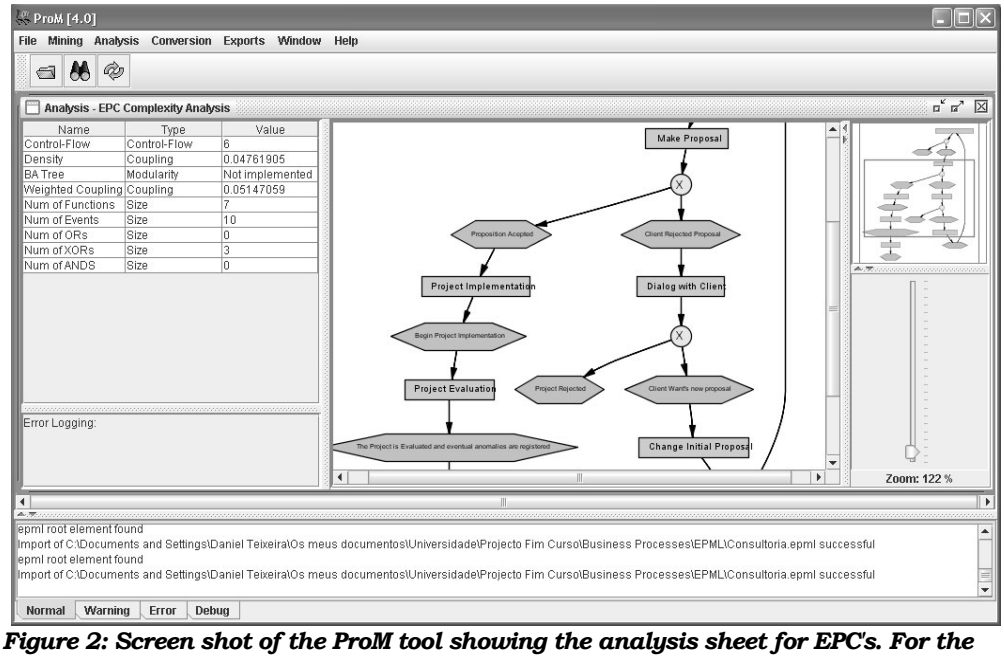

Figure 2: Screen shot of the ProM tool showing the analysis sheet for EPC's. For the EPC process model presented, several metrics are calculated. Note that this tool is still under development and that some concepts on the screen shot (e.g. BA Tree, and Weighted Coupling) are not explained in this chapter.

As can be seen from the figure, the size of a process model has several components, such as the number of events, functions, ORs, XORs, and ANDs. Events and functions are specific elements to EPCs. The figure also shows the Control-Flow Complexity (Cardoso, 2005a) of the process displayed which is 6, the density (Mendling, 2006) which is 0.048, and the weighted coupling which is 0.515. While, at this

point in time, these numbers may be rather complicated to interpret for someone outside this area of research, we expect that when organizations have successfully implemented quality metrics as part of their process development projects, empirical results and practical results from real world implementation will set limits and threshold for processes. Recall that this scenario happened with the McCabe cyclomatic complexity (Frappier, Matwin, & Mili, 1994).

Data oriented Cohesion and Coupling in ProM

Within the ProM framework, also an environment is developed to calculate cohesion and coupling metrics based on the theory in (Reijers & Vanderfeesten, 2004). The proposed coupling-cohesion ratio can be used to compare alternative designs defined on the same PDM. In this context, a design is a grouping of data elements and their respective operations into activities, such that every activity contains one or, preferably, more operations. The best design is the one with the lowest coupling-cohesion ratio.

In Figure 2 and Figure 3, some screen shots of the cohesion and coupling environment are shown. Both screen shots contain a different design based on the same PDM. In the first design (Figure 2) the process cohesion value is 0.183, the process coupling value is 0.714, and the coupling-cohesion ratio is 3.902. The part of activity A is also indicated in the PDM. For the second design (Figure 3) these values are: 0.123, 0.867, and 7.049. When comparing the values for the coupling-cohesion ratio for both designs we see that the first design has a lower ratio and thus is the better alternative of the two process designs.

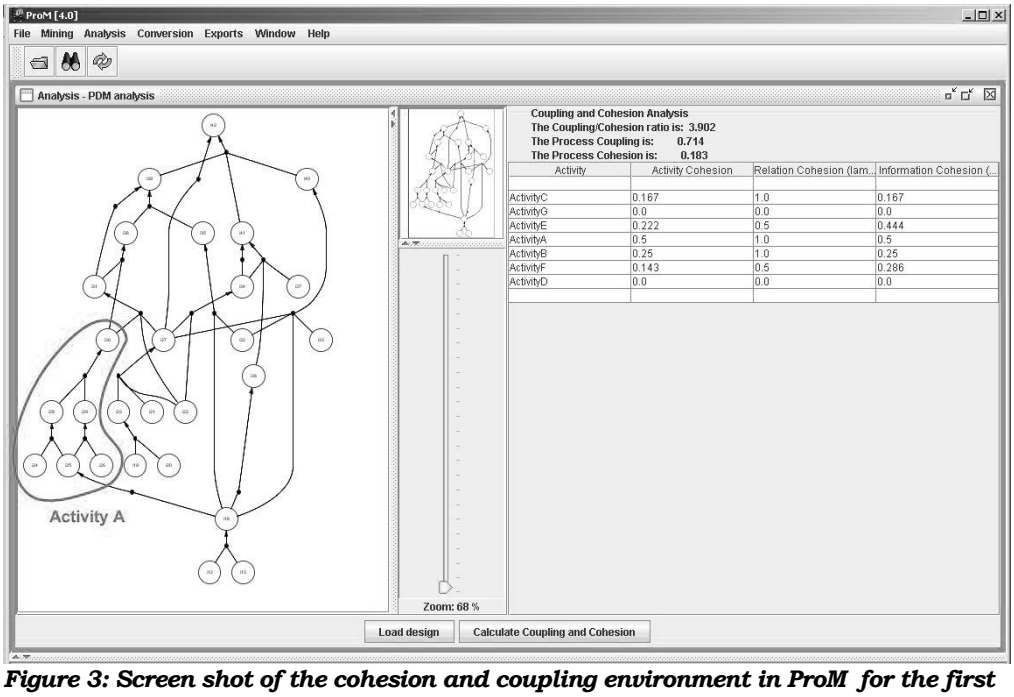

Figure 3: Screen shot of the cohesion and coupling environment in ProM for the first design. This design contains seven activities that are defined on the PDM. Activity A is indicated.

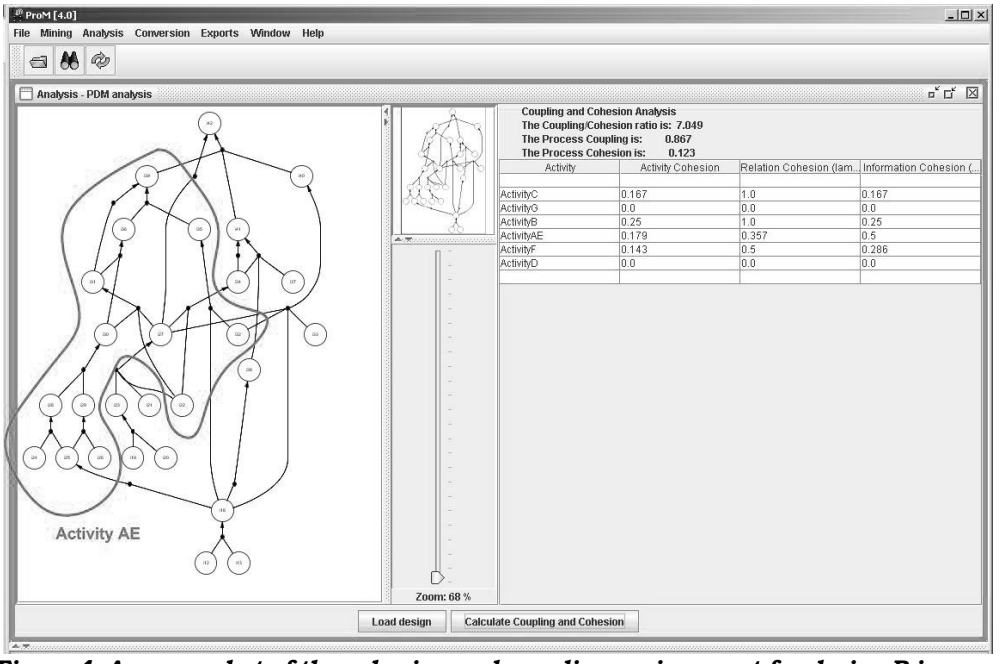

Figure 4: A screen shot of the cohesion and coupling environment for design B in ProM. Design B contains six activities that are defined on the PDM and it differs from design A because activities A and E are merged. Activity AE is indicated in the PDM in the screen shot.

CONCLUSION

Currently, organizations are modeling and designing business processes without the aid of metrics to question the quality or properties of their models. As a result, it may happen that simple processes are modeled in a complex and unsuitable way. This may lead to a lower understandability, higher maintenance costs, and perhaps inefficient execution of the processes in question (e.g. when such models are used to enact). Considering the efforts that modern organizations spend on creating and maintaining business processes we can truly speak of a great opportunity for the use of quality metrics here.

Examples of important questions that can be made relative to a process model are: "can process P1 be designed in a simpler way?", "what is the complexity of process P2?". "is process P3 difficult to adapt?" and "can process P4 be easily restructured into sub-processes?" In the future, these kinds of questions can perhaps be satisfactorily answered with the use of process metrics such as coupling, cohesion, complexity, modularity, and size metrics. Each of these metrics analyses a business process from a particular perspective.

It is clear that quality metrics for business processes need yet to come to full bloom. In particular, much empirical work needs to be done to assess the applicability and validity of the various proposed metrics. However, both for practitioners and researchers there is highly attractive prospect of sophisticated tools coming available that are capable to thoroughly analyze process models against low cost, at considerable speed, and yielding tangible business benefits.

ACKNOWLEDGEMENT

This research is partly supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Dutch Ministry of Economic Affairs.

REFERENCES

Aalst, W.M.P. van der (1999). On the automatic generation of workflow processes based on product structures. Computers in Industry, 39, 2, pp. 97-111.

Aalst, W.M.P. van der; Weijters, A.J.M.M.; and Maruster, L. (2004). Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering, 16(9), pp.1128–1142.

Baresi, L.; Casati, F.; Castano, S.; Fugini, M.; Mirbel, I.; Pernici, B.; and Pozzi, G. (1999). Workflow Design Methodology. In P. Grefen, B. Pernicii and G. Sanchez, editors, Database Support for Workflow Management: the WIDE Project, pp. 47-94, Kluwer Academic Publishers.

Bieman, J.M., and Kang, B.-K. (1998). Measuring Design-level Cohesion. IEEE Transactions on Software Engineering, 24, 2, pp. 111-124.

Brandes, U., and Erlebach, T., editors (2005). Network Analysis: Methodological Foundations [outcome of a Dagstuhl seminar, 13-16 April 2004], volume 3418 of Lecture Notes in Computer Science. Springer-Verlag.

Card, D.N.; Church, V.E.; and Agresti, W.W. (1986). An Empirical Study of Software Design Practices. IEEE Transactions on Software Engineering, 12, 2, pp. 264-271.

Cardoso, J. (2005a). How to Measure the Control-flow Complexity of Web Processes and Workflows. In: Fischer, L., ed., Workflow Handbook 2005, pp. 199-212, Lighthouse Point.

Cardoso, J. (2005b). Control-flow Complexity Measurement of Processes and Weyuker's Properties. Proceedings of the 6th International Enformatika Conference (IEC 2005), International Academy of Sciences, Budapest, Hungary. Vol. 8pp. 213-218.

Cardoso, J. (2006). Process control-flow complexity metric: An empirical validation, IEEE International Conference on Services Computing (IEEE SCC 06), Chicago, USA, pp. 167-173, IEEE Computer Society.

Cardoso, J.; Mendling, J.; Neuman, G. & Reijers, H.A. (2006). A discourse on complexity of process models. In: Eder, J.; Dustdar, S. et al, editors, BPM 2006 workshops. Lecture Notes in Computer Science 4103, Springer-Verlag, Berlin, pp. 115-126.

Conte, S.D.; Dunsmore, H.E.; and Shen, V.Y. (1986). Software Engineering Metrics and Models, Benjamin/Cummings Publishing Company, Inc..

Dongen, B.F. van; Alves de Medeiros, A.K.; Verbeek, H.M.W. ; Weijters, A.J.M.M.; and Aalst, W.M.P. van der (2005). The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, Application and Theory of Petri Nets 2005, volume 3536 of Lecture Notes in Computer Science, pp. 444–454, Springer-Verlag, Berlin.

Frappier, M., Matwin, S. and Mili. A. (1994). Software Metrics for Predicting Maintainability: Software Metrics Study: Technical Memorandum 2. Canadian Space Agency, January 21.

Gruhn, V., and Laue, R. (2006). Complexity metrics for business process models. In: Witold Abramowicz and Heinrich C. Mayer, editors, 9th international conference on business information systems (BIS 2006), vol. 85 of Lecture Notes in Informatics, pp. 1-12.

Gucegliogl u, A.S., and Demiros, O.W. (2005). Using Software Quality Characteristics to Measure Business Process Quality. In W.M.P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, editors, Business Process Management (BPM 2005), Lecture Notes in Computer Science, volume 3649, pages 374-379, Springer-Verlag, Berlin.

Jones, T. C. (1986). Programming Productivity. New York, McGraw-Hill.

Kang, B.-K., and Bieman, J.M. (1996). Using Design Cohesion to Visualize, Quantify, and Restructure Software. 8th International Conference on Software Engineering and Knowledge Engineering, Knowledge Systems Institute, Skokie IL, pp. 222-229.

Kang, B.-K., and Bieman, J.M. (1999). A Quantitative Framework for Software Restructuring. Journal of Software Maintenance, 11, pp. 245-284.

Keller, G., and Teufel, T. (1998). SAP(R) R/3 Process Oriented Implementation: Iterative Process Prototyping. Addison-Wesley.

Kostoff, R.N. (1999). Science and Technology Innovation. Technovation, 19, 10, pp. 593-604.

Latva-Koivisto, A.M. (2001). Finding a complexity measure for business process models. Helsinki University of Technology, Systems Analysis Laboratory.

Mendling, J. (2006). Testing Density as a Complexity Metric for EPCs. Technical Report JM-2006-11-15. Vienna University of Economics and Business Administration. Retrieved from http://wi.wu-wien.ac.at/home/mendling/publications/TR06-density.pdf

Mendling, J.; Moser, M.; Neumann, G.; Verbeek, H.M.W.; Dongen, B.F. van; and Aalst, W.M.P. van der (2006a). A Quantitative Analysis of Faulty EPCs in the SAP Reference Model. BPM Center report BPM-06-08, Eindhoven University of Technology, Eindhoven.

Mendling, J.; Moser, M.; Neumann, G.; Verbeek, H.M.W.; Dongen, B.F. van; and Aalst, W.M.P. van der (2006b). Faulty EPCs in the SAP Reference Model. In: J.L. Fiadeiro, S. Dustdar and A. Sheth, editors, Proceedings of BPM2006, Lecture Notes in Computer Science, volume 4102, pp. 451-457, Springer-Verlag, Berlin.

Myers, G.J. (1978). Composite/Structured Design. Van Nostrand Reinhold, New York, NY.

Reijers, H.A. (2003). Design and control of workflow processes: business process management for the service industry. Lecture Notes in Computer Science 2617, Springer-Verlag, Berlin.

Reijers, H.A. (2003). A cohesion metric for the definition of activities in a workflow process. Proceedings of the 8th CAiSE/IFIP8.1 International workflop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD 2003), pp. 116-125.

Reijers, H.A.; Limam Mansar, S.; and Aalst, W.M.P. van der (2003). Product-Based Workflow Design. Journal of Management Information Systems, 20, 1, pp. 229-262.

Reijers, H. A., and Vanderfeesten, I.T.P. (2004). Cohesion and Coupling Metrics for Workflow Process Design. In J. Desel, B. Pernici, and M. Weske, editors, Proceedings of the 2nd International Conference on Business process Management (BPM 2004), Lecture Notes in Computer Science volume 3080, pp. 290-305, Springer-Verlag, Berlin.

Rozinat, A., and Aalst, W.M.P. van der (2006). Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models. In C. Bussler et al, editor, BPM 2005 Workshops (Workshop on Business Process Intelligence), volume 3812 of Lecture Notes in Computer Science, pp. 163-176, Springer-Verlag, Berlin.

Selby, R.W., and Basili, V.R. (1991). Analyzing Error-Prone System Structure. IEEE Transactions on Software Engineering, 17, 2, pp. 141-152.

Shen, V.Y.; Yu, T.-J.; Thebaut, S.M.; and Paulsen, L.R. (1985). Identifying Error-Prone Software, IEEE Transactions on Software Engineering, 11, 4, pp. 317-324.

Shepperd, M. (1993). Software Engineering Metrics Volume I: Metrics and Validations, McGraw-Hill.

Troy, D.A., and Zweben, S.H. (1981). Measuring the Quality of Structured Designs, Journal of Systems and Software, vol. 2, pp. 113-120.

Weyuker, E.J. (1988). Evaluating Software Complexity Measures. IEEE Transactions on Software Engineering, 14, 9, pp. 1357-1365.